

ISSN



9 772903 075009



# Jurnal Teknik Informatika

Desember 2013

Vol. I #1

Rekursif

Created with



**nitro**

PDF<sup>®</sup>

professional

download the free trial online at [nitropdf.com/professional](http://nitropdf.com/professional)

download the free trial online at [nitropdf.com/professional](http://nitropdf.com/professional)

## PENGENALAN POLA AKSARA KA-GA-NGA DENGAN METODE *LEARNING VECTOR QUANTIZATION* (LVQ)

Naimah Lubis, Edy Hermansyah dan Desi Andreswari  
Program Studi Teknik Informatika, Fakultas Teknik, Universitas Bengkulu  
Jl. WR. Supratman, Kandang Liman Bengkulu 38371 A  
desiandreswari@yahoo.co.id

### ABSTRACT

Ka-Ga-Nga is ancient script which owned by Rejang and Serawai ethnic living in Bengkulu region. It has 28 characters, each character followed by punctuation marks (*sandangan*). Ka-Ga-Nga is much more complicated compared to Roman's alphabets. Each character has a lot of similarity and can be differentiated by only a stretch line. That is why process to recognize Ka-Ga-Nga pattern is considered to having high difficulty level. The research not only will study recognizing Ka-Ga-Nga in handwritten form, but also recognizing Ka-Ga-Nga in manuscripts. The manuscripts will be tested has a lot of noise. Therefore, a complex image processing is needed to recognize character in manuscripts. Moreover, building a successful Ka-Ga-Nga character recognition system, cannot be fulfilled without solving the segmentation problem. Segmentation is process to separates characters in the text. In this research, the segmentation method that used for resulted in some combining operations of morphology. This method gives a good result for script with big space. Then, for the classification process used Learning Vector Quantization (LVQ) method. By using LVQ to build system, we have achieved a good accuracy level by combining maximum epoch with big value and learning rate with the value near zero.

*Key words : pattern recognition, learning vector quantization, image processing*

### 1. PENDAHULUAN

Bengkulu merupakan salah satu daerah di Indonesia yang memiliki kekayaan peninggalan masa lalu. Salah satunya adalah peninggalan dalam bentuk tulisan berupa naskah kuno. Naskah kuno yang terdapat di Bengkulu kebanyakan ditulis dalam aksara Ka-Ga-Nga.

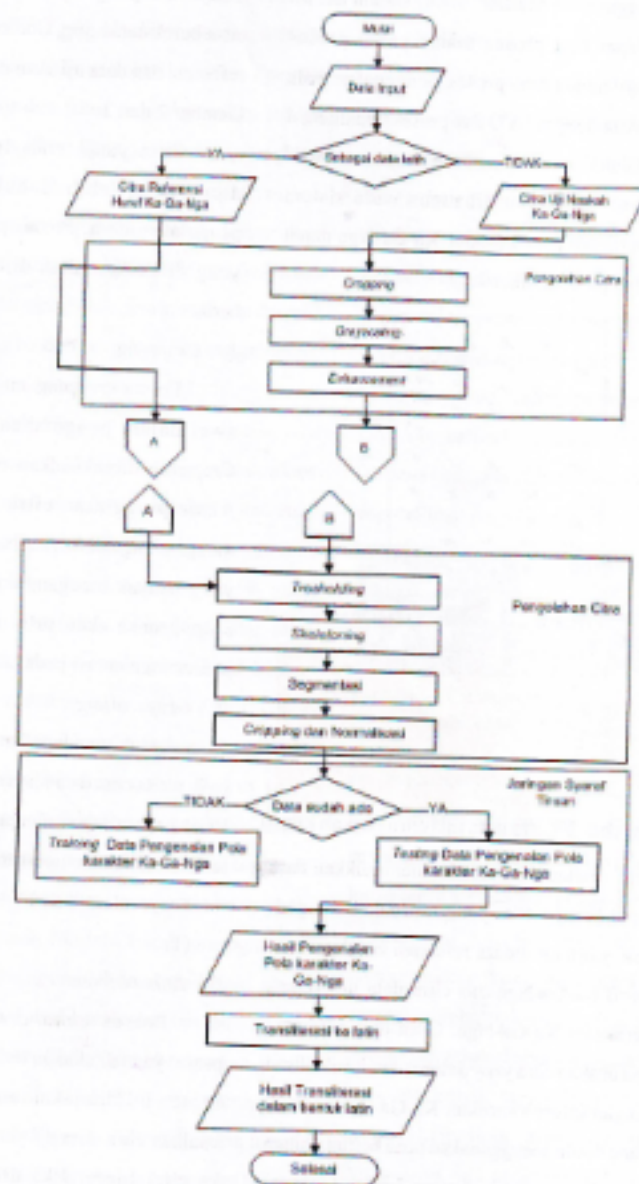
Sarwit Sarwono dalam Laporan Penelitian Hibah Bersaing [1] menyatakan bahwa dari sejumlah suku yang ada di Bengkulu hanya beberapa kalangan orang tua yang masih dapat membaca aksara ulu/Ka-Ga-Nga. Selain itu kondisi fisik naskah sebagian besar masih berada di tangan masyarakat sehingga rentan mengalami kerusakan. Jika tidak ada upaya pelestarian, maka naskah kuno akan terancam punah.

Salah satu cara untuk mengatasi hal tersebut adalah dengan melakukan digitalisasi. Digitalisasi dan transliterasi akan lebih mudah dilakukan dengan adanya sistem yang dapat mengenali setiap karakter aksara Ka-Ga-Nga.

Ada banyak pendekatan algoritma yang dapat digunakan untuk mengembangkan sistem pengenalan pola karakter, salah satunya adalah *neural network*. Dalam penelitian ini akan digunakan jaringan syaraf tiruan metode *Learning Vector Quantization* (LVQ) yang merupakan salah satu jenis Jaringan Syaraf Tiruan *Supervised Learning* pada lapisan kompetitif untuk proses klasifikasi karakter Ka-Ga-Nga. Hasil yang ingin dicapai adalah bentuk latin dari karakter yang terdapat dalam naskah agar

memudahkan penerjemahan ke dalam bahasa Indonesia. Maka hasil dari proses klasifikasi

akan dilanjutkan dengan proses transliterasi karakter ke dalam bentuk latin.



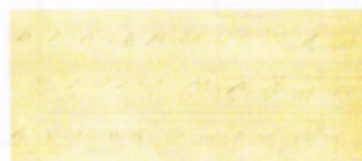
Gambar 1. Flowchart sistem pengenalan pola aksara Ka-Ga-Nga dengan model Learning Vector Quantisation (LVQ)

## 2. METODE PENELITIAN

Sistem yang akan dibangun menggunakan Matlab® 7.7.0. Dalam hal ini terdapat tiga proses utama yakni proses pengolahan citra, proses pengenalan pola karakter dengan LVQ dan proses transliterasi ke dalam bentuk latin. Tahapan perancangan sistem yang akan dibangun pada sistem pengenalan pola karakter Ka-Ga-Nga dapat dilihat pada Gambar 1.



Gambar 2. Citra data referensi



Gambar 3. Citra data uji (citra naskah kuno)

Tahap awal adalah memasukkan data input berupa citra yang dibedakan menjadi dua yaitu citra data referensi untuk sampel huruf Ka-Ga-Nga dan citra data uji naskah beraksara Ka-Ga-Nga. Citra data referensi merupakan citra yang didapat dari hasil tulisan tangan seluruh karakter Ka-Ga-Nga serawai yang ditulis menggunakan pena bertinta hitam pada kertas putih. Setiap kertas terdapat sekitar 50 karakter Ka-Ga-Nga. Kemudian dilakukan proses *scanning* menggunakan

*scanner* Canon MP258 dengan resolusi sebesar 100 dpi. Sedangkan citra data uji didapat dari pengumpul naskah kuno dengan citra berekstensi .jpg. Contoh dari citra data referensi dan data uji akan ditampilkan pada Gambar 2 dan 3.

Citra yang telah diinputkan akan diproses terlebih dahulu pada tahap pengolahan citra. Proses pengolahan citra yang dilakukan akan dijelaskan sebagai berikut:

### 1. *Cropping*

Proses *cropping* merupakan langkah awal dalam pengolahan citra data uji. *Cropping* dimaksudkan untuk memotong *frame*/pinggiran citra yang banyak mengandung *noise*. Jika bagian pinggir citra yang banyak mengandung *noise* tidak di-*crop*, maka akan sulit untuk melakukan proses segmentasi pada tahap selanjutnya.

### 2. *Grayscale*

Untuk mendapatkan citra biner yang baik, maka citra data uji yang merupakan citra RGB harus diubah menjadi citra *grayscale* terlebih dahulu dengan rumus [2]:

$$\text{Grayscale} = 0.299R + 0.587G + 0.114B \quad (1)$$

### 3. *Enhancement*

Proses *enhancement* merupakan proses yang dilakukan terhadap citra data uji. Proses ini ditujukan untuk memperbaiki kualitas citra data uji yang akan dikonversi ke citra biner. Jika dilakukan konversi langsung dari citra hasil *grayscale* ke citra biner maka citra data uji yang didapatkan



masih memiliki banyak *noise*. Sedangkan jika melewati tahap filtering maka karakter huruf ka-Ga-Nga yang terdapat pada data citra uji lebih terlihat jelas. Pada proses ini, metode yang digunakan yakni *Linear Spatial Filtering* dan filter yang digunakan adalah *highpass filter*. Penggunaan *highpass filter* akan sangat sesuai digunakan untuk mempertajam tepi citra. *Highpass filter* akan mempertajam komponen dengan frekuensi tinggi dan menghilangkan komponen dengan frekuensi rendah sehingga akan sangat cocok untuk penajaman tepi citra [3].

#### 4. Thresholding

Proses *thresholding* merupakan proses untuk mendapatkan citra biner. Ada banyak metode yang digunakan dalam *thresholding* citra. Dalam sistem ini yang digunakan adalah metode Otsu. Jika menggunakan metode *threshold* biasa, dengan rentang nilai antara 0-1 tidak didapatkan citra biner dengan kualitas gambar yang baik atau yang sesuai dengan yang diharapkan. Pada tahap ini juga dilakukan operasi negasi pada citra karena terdapat perbedaan nilai piksel untuk citra biner pada pemrograman Matlab. Pada pemrograman Matlab, komponen berwarna hitam akan bernilai nol dan komponen berwarna putih bernilai satu. Pada citra, komponen yang berwarna putih (nilai piksel satu) merupakan komponen *background* dan berwarna hitam (nilai piksel nol) merupakan komponen tulisan karakter Ka-Ga-Nga. Operasi negasi yang dilakukan akan menjadikan komponen

*background* berwarna hitam (nilai piksel nol) dan komponen karakter berwarna putih (nilai piksel satu).

#### 5. Skeletons

Proses *skeletons* bertujuan untuk menghasilkan objek yang memiliki ketebalan satu piksel atau dengan kata lain mengekstraksi kerangka objek. Hal ini dilakukan untuk mengatasi perbedaan nilai bobot yang akan dihasilkan dalam proses pengenalan pola menggunakan *Learning Vector Quantization* untuk karakter dalam kelas yang sama. Karena pada sistem ini, citra masukan berasal dari hasil pemulisan yang berbeda-beda. Perubahan citra hasil *skeletons* dapat dilihat pada Gambar 5.



Gambar 4. Citra tanpa *skeletons*



Gambar 5. Citra hasil *skeletons*

#### 6. Segmentasi

Proses segmentasi bertujuan untuk mendapatkan area-area yang merupakan karakter Ka-Ga-Nga dalam citra. Proses segmentasi yang dilakukan akan menggunakan kombinasi dari beberapa operasi morfologi. Morfologi merupakan operasi pengekstraksian komponen citra yang berguna untuk merepresentasikan dan mendeskripsikan area bentuk seperti *boundaries* dan *skeletoning* [4].

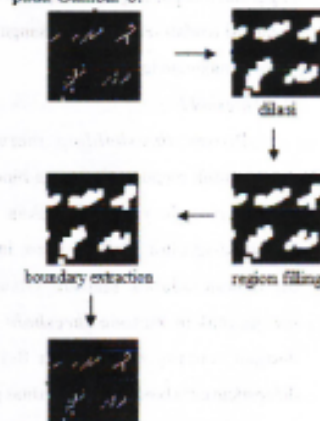
Operasi morfologi matematika yang digunakan adalah operasi dilasi, *region filling*,

*extraction of connected component*, dan *boundary extraction*. Tahap awal operasi morfologi matematika adalah menentukan *structuring element* (SE). SE merupakan matrik konvolusi, SE yang akan digunakan pada proses ini adalah SE *square* yakni matriks yang memiliki ukuran kolom dan baris yang sama atau lebih dikenal dengan matriks  $m \times m$  dengan setiap elemen bernilai 1 untuk semua baris dan kolom. Penentuan besar kecilnya ukuran matriks SE yang digunakan akan mempengaruhi hasil dilasi. Setelah menentukan ukuran SE, operasi dilasi akan dapat dilakukan. Alasan mengapa operasi dilasi ini dilakukan pada proses segmentasi adalah untuk memperbesar ukuran objek (*foreground*) sebelum dilakukan pelabelan agar objek dengan tanda baca tersegmentasi menjadi 1 karakter. Ukuran piksel dari citra tidak mengalami perubahan.

Operasi morfologi setelah dilasi adalah *region filling*. Operasi ini akan memblok semua lubang-lubang yang berada di tengah objek dengan cara menelusuri nilai piksel yang memiliki keterhubungan dengan piksel yang berada di dekatnya atau yang lebih dikenal dengan *connected neighborhood element*. Pada sistem ini akan digunakan 4-*connected neighborhood element*.

Setelah operasi *region filling*, operasi morfologi yang akan dilakukan selanjutnya adalah *extraction of connected component*. Pada operasi ini akan terjadi pelabelan area objek (*foreground*) pada objek yang elemennya memiliki keterhubungan. Setelah

diberikan label pada setiap objek, selanjutnya dilakukan pembatasan area tiap objek dengan menggunakan operasi *boundary extraction*. *Boundary extraction* merupakan operasi pembatasan yang akan menentukan area dari tiap objek/karakter pada tahap segmentasi. Pembatasan area objek dilakukan dengan membuat *boundary* berbentuk *rectangle* untuk setiap objek/karakter. Seluruh operasi yang ada pada proses segmentasi ditampilkan pada Gambar 6.



Gambar 6. Proses segmentasi

## 7. Cropping dan Normalisasi

Proses ini merupakan proses memotong satu bagian objek/karakter dari citra hasil segmentasi. Kemudian setiap citra karakter yang telah di-crop akan dinormalisasi dengan ukuran 20x20 piksel.

Setelah mendapatkan nilai piksel dari objek yang akan dikenali dari tahap pengolahan citra digital maka tahap selanjutnya adalah melakukan analisis penyelesaian teknik jaringan syaraf tiruan. metode jaringan syaraf tiruan yang dipilih

adalah *Learning Vector Quantization* (LVQ). LVQ adalah suatu metode untuk melakukan pembelajaran pada lapisan kompetitif yang terawasi. Suatu lapisan kompetitif akan secara otomatis belajar untuk mengklasifikasikan vektor-vektor *input*. Jika 2 vektor *input* mendekati sama, maka lapisan kompetitif akan meletakkan kedua vektor *input* tersebut ke dalam kelas yang sama [5]. Adapun algoritma LVQ adalah sebagai berikut [5]:

1. Tetapkan bobot ( $w$ ), Maximum epoch (MaxEpoch), *error* minimum yang diharapkan (Eps), *Learning rate* ( $\alpha$ ).
2. Masukan :
  - *Input* :  $x(m,n)$ ;  $m$ =jumlah *input*;  $n$ =jumlah data
  - Target :  $T(1,n)$
3. Tetapkan kondisi awal :
  - epoch = 0
  - err = 1
4. Kerjakan jika (epoch < MaxEpoch) atau ( $\alpha$  > eps)
  - a. Epoch = epoch + 1
  - b. Kerjakan untuk  $i=1$  sampai  $n$ 
    - i. Tentukan  $J$  hingga  $\|x-w_j\|$  minimum (sebut sebagai  $C_j$ )
    - ii. Perbaiki  $w_j$  dengan ketentuan :
      - Jika  $T = C_j$  maka :  $w_j(\text{baru}) = w_j(\text{lama}) + \alpha[x - w_j(\text{lama})]$
      - Jika  $T \neq C_j$  maka :  $w_j(\text{baru}) = w_j(\text{lama}) - \alpha[x - w_j(\text{lama})]$
    - c. Kurangi nilai  $\alpha$

Keterangan notasi :

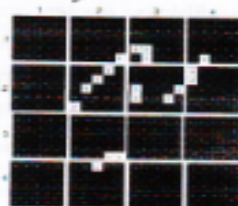
$x$  = vektor pelatihan ( $x_1 \dots x_i \dots x_n$ )

$T$  = kategori yang benar untuk vektor pelatihan

$w_j$  = vektor bobot unit *output*  $j$  ( $w_{1j} \dots w_{ij} \dots w_{nj}$ )

$C_j$  = kategori yang diwakili oleh unit *output*  $j$

$\|x-w_j\|$  = jarak Euclidean antara vektor *input* dan vektor bobot untuk unit *output*  $j$ .



Gambar 7. Membagi piksel karakter ke dalam matriks 4x4

Tabel 1. Tingkat akurasi *training* dan jumlah *epoch* yang berbeda

Jumlah Epoch	Learning rate ( $\alpha$ )	Akurasi ( <i>training</i> )
10	0,2	92,5%
10	0,15	92,5%
10	0,1	92,5%
10	0,05	92,5%
20	0,2	95%
20	0,15	95%
20	0,1	95%
20	0,05	95%
30	0,2	95%
30	0,15	95%
30	0,1	95%
30	0,05	95%
40	0,2	95%
40	0,15	95%
40	0,1	95%
40	0,05	95%

Dalam metode LVQ terdapat dua proses utama yakni:

#### 1. Proses *Training*

Proses *training* ini bertujuan untuk membuat data referensi pada *neural network*. Pada tahap ini 50 citra karakter dari data latih akan diklasifikasikan ke dalam 10 kelas. Jadi akan terdapat 10 vektor yang akan diinisialisasikan sebagai bobot awal.



Sedangkan data yang akan dilatih sebanyak 40 data. Vektor input yang dihasilkan pada tahap normalisasi adalah 20x20 piksel. Hal ini akan mengakibatkan penggunaan memori yang besar. Untuk mengatasi hal tersebut, maka akan dilakukan *blocking* citra dengan menggunakan metode *nonoverlapping* citra. Setiap citra akan dibagi ke matriks berukuran 4x4. Sehingga masing-masing elemen matriks berukuran 5x5 piksel seperti yang disajikan pada Gambar 7.

Setiap blok matriks akan diberi nilai sesuai dengan perhitungan rata-rata nilai elemen pada tiap blok. Untuk citra pada Gambar 7, masing-masing blok akan bernilai:



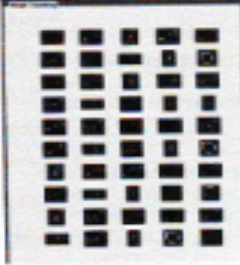
	1	2	3	4
1	0	0.04	0.12	0.04
2	0	0.16	0.16	0.08
3	0	0.08	0	0
4	0	0.04	0	0

Maka karakter pada Gambar 7 akan memiliki vektor bobot (0 0.04 0.12 0.04 0 0.16 0.08 0 0 0 0.04 0 0).

## 2. Proses *Testing*

Setelah dilakukan proses *training* maka proses selanjutnya adalah proses *testing*. Proses ini dilakukan setelah proses *training* berakhir. *Testing* akan dilakukan terhadap teks kuno Ka-Ga-Nga dengan membandingkan jarak *input* terhadap seluruh bobot dari proses *training*. Nomor dari bobot dengan jarak terkecil akan menjadi kelas dari karakter data uji.

Tabel 2. Hasil transliterasi masing-masing citra data uji

Jenis data	Citra karakter hasil pengujian	Hasil transliterasi
Belum pernah dilatih (Naskah kuno 4a.jpg)		ti ti pa nga pa ka nga ra ra nga ra nga nta ma po pa nta nta ra ti ka ma nta pa ka ma nta ma ma ti pa nta ma ti . nta ka nta ka ti nta nta ma ra ti
Belum pernah dilatih (hasil tulisan dari penulis yang sama)		ti ti pa ti ti
Citra data latih		ngan ti ka ti pa ma ra nta ma ngan ti ka ngan ti ma nta ra ma ma ti ma ngo ngan ma pa nta ra ma ka ti nga ngan po ma nta ma ra . ka ma nga ngan pa nta ma ma . ra



Setelah citra berhasil dikenali, selanjutnya dilakukan transliterasi setiap karakter dengan menggunakan bantuan kode ASCII desimal.

Setelah semua proses berhasil dilakukan maka akan diketahui tingkat akurasi yang dihasilkan sistem. Untuk tingkat akurasi sistem pada proses *testing* dirumuskan sebagai berikut:

$$\text{akurasi}(\%) = \frac{n_t}{N} \times 100\% \quad (2)$$

dengan,

$n_t$  = jumlah karakter yang benar sesuai dengan target Intih

$N$  = jumlah seluruh karakter data latih

untuk tingkat akurasi proses testing dapat dirumuskan sebagai berikut:

$$\text{akurasi}(\%) = \frac{n_{te}}{N} \times 100\% \quad (3)$$

$n_{te}$  = jumlah karakter yang dikenali dengan benar dibandingkan dengan hasil sebelum diuji

$N$  = jumlah seluruh karakter data uji

### 3. HASIL DAN PEMBAHASAN

Dari proses *training* terhadap citra data latih pada Gambar 2 didapat tingkat akurasi *training* yang berbeda untuk beberapa jumlah *epoch* dan nilai *learning rate* dapat dilihat pada Tabel 1.

Dapat dilihat pada Tabel 1 bahwa semakin besar jumlah *epoch* maka akan menghasilkan *error* yang semakin kecil atau tingkat akurasi yang semakin besar. Setelah *training* maka akan dilakukan pengujian terhadap data yang belum pernah dilatih berupa citra naskah kuno, data yang belum pernah dilatih dari penulis huruf yang sama dan data yang pernah dilatih. Tabel 2 akan

menampilkan hasil transliterasi untuk setiap data uji.

Tabel 3. Tingkat akurasi hasil *testing* citra data uji yang belum pernah dilatih (naskah 4a.jpg)

Jumlah Epoch	Learning rate (α)	Akurasi <i>testing</i>
10	0,2	20%
10	0,15	20%
10	0,1	20%
10	0,05	20%
20	0,2	20%
20	0,15	20%
20	0,1	20%
20	0,05	20%
30	0,2	20%
30	0,15	20%
30	0,1	20%
30	0,05	20%
40	0,2	20%
40	0,15	20%
40	0,1	20%
40	0,05	20%

Tabel 4. Tingkat akurasi hasil *testing* citra data uji yang belum pernah dilatih (hasil tulisan dari penulis yang sama)

Jumlah Epoch	Learning rate (α)	Akurasi <i>testing</i>
10	0,2	80%
10	0,15	80%
10	0,1	80%
10	0,05	100%
20	0,2	80%
20	0,15	80%
20	0,1	80%
20	0,05	100%
30	0,2	80%
30	0,15	80%
30	0,1	80%
30	0,05	100%
40	0,2	80%
40	0,15	80%
40	0,1	80%
40	0,05	100%

Hasil akurasi sistem untuk masing-masing data akan ditampilkan dalam Tabel 3, Tabel 4 dan Tabel 5.

Berdasarkan hasil pada Tabel 3, 4 dan 5 didapat kesimpulan bahwa jumlah *epoch*

dan *learning rate* juga ikut mempengaruhi hasil *testing*. Pada tabel tersebut akurasi yang dihasilkan sistem untuk setiap *epoch* tidak mengalami perubahan yang cukup besar, sedangkan *learning rate* akan mempengaruhi hasil akurasi sistem jika dipilih nilai terkecil. Jadi untuk mendapatkan tingkat error yang mendekati nilai 0 atau tingkat akurasi mendekati 100% dapat digunakan *epoch* yang besar dan nilai *learning rate* yang kecil.

Tabel 5. Tingkat akurasi hasil *testing* citra data uji dari data yang pernah dilatih

Jumlah Epoch	Learning rate ( <i>n</i> )	Akurasi <i>testing</i>
10	0,2	88%
10	0,15	88%
10	0,1	88%
10	0,05	86%
20	0,2	90%
20	0,15	90%
20	0,1	90%
20	0,05	92%
30	0,2	90%
30	0,15	90%
30	0,1	90%
30	0,05	92%
40	0,2	90%
40	0,15	90%
40	0,1	90%
40	0,05	92%

#### 4. KESIMPULAN

Dari penelitian dapat disimpulkan bahwa metode *Learning Vector Quantization* (LVQ) dapat digunakan untuk mengenal pola karakter Ka-Ga-Nga hasil tulisan tangan. Dengan data referensi yang terbatas, sistem masih mampu mengenali karakter Ka-Ga-Nga hasil tulisan tangan. Hasil terbaik jika data yang diujikan merupakan data yang pernah dilatih dan data yang bukan data latih tetapi hasil penulisan

oleh penulis yang sama dengan data latih. Sedangkan untuk mengenali tulisan yang ada di naskah kuno tingkat akurasi yang dihasilkan sistem masih kecil. Pemilihan jumlah *epoch* dan nilai *learning rate* akan mempengaruhi tingkat akurasi sistem, yakni:

1. Semakin besar jumlah *epoch* maka tingkat akurasi *training* sistem akan mendekati 100%.
2. Semakin besar jumlah *epoch* dan semakin kecil nilai *learning rate* maka tingkat akurasi *testing* sistem akan semakin besar.

Berdasarkan hasil penelitian, pengujian serta pembahasan yang dibahas, maka disarankan untuk pengembangan penelitian dimasa yang akan datang dapat mencoba model *neural network* yang lain sehingga menghasilkan akurasi yang lebih baik dibandingkan dengan metode LVQ. Selain itu, dapat dikembangkan sistem yang bisa melakukan transliterasi untuk setiap kalimat.

#### DAFTAR PUSTAKA

- [1]. Sarwono, S dkk., 2007, Pemetaan Penulis dan Pusat Penulisan Naskah-Naskah Ulu Melalui Penelusuran Naskah-Naskah Ulu Pada Masyarakat di Provinsi Bengkulu, Laporan Penelitian Hibah Bersaing, Proyek Multitahun, Dikti, Jakarta.
- [2]. Munir, R., 2004, *Pengolahan Citra*

- [3]. Putri, D., 2010, **Pengolahan Citra Digital**, Andi Offset, Yogyakarta.
- [4]. Gonzalez RC, Woods RE, 2004, **Digital Image Processing**, Prentice Hall, New Jersey.
- [5]. Kusumadewi, S., 2003, **Artificial Intelligence (Teknik dan Aplikasinya)**, Graha Ilmu, Jakarta.